Recap of the first day' s lecture on Substitution Ciphers

At its core, cryptography is about making encryption systems that can keep secrets away from smart and wise attackers who try to crack them. A famous NSA maxim states that attacks never get worse, they only get better.

Today we discussed a very simple method of encryption. It is too simple and weak to use in practice -- after all, we were able to crack it very quickly. Why do we mention it, then? Moreover, why will continue to academically study weak cryptosystems? Because they demonstrate a number of useful principles about what is needed for a secure system.

We studied substitution ciphers like one sees in the comic section. More properly, these are called “monoalphabetic substitution ciphers”. Here one simply replaces each occurence of a given letter of the alphabet by another letter, which is dictated by a key. I will now demonstrate this with an example (not the same one I used in class) and some sample mathematica code. It will take some plaintext, e.g., I’ll simply use the first paragraph above:

```
In[9]:= plaintext = "At its core,
        cryptography is about making encryption systems that can keep secrets away
        from smart and wise attackers who try to crack them.
        A famous NSA maxim states that attacks never get worse,
        they only get better.
    "

Out[9]= At its core, cryptography is about making encryption systems that can keep
        secrets away from smart and wise attackers who try to crack them.  A
        famous NSA maxim states that attacks never get worse, they only get better.
```

Next is a command which converts each character in my text into ASCII, a common computer code that represents alphanumeric characters as numbers. It is easier to manipulate these numbers. First I convert all letters from upper case to lower case, but keep punctuation (which I could also get rid of, but for now I’d like to keep it in).

```
In[10]:= numbersofplaintext = ToCharacterCode[ToLowerCase[plaintext]]

Out[10]= {97, 116, 32, 105, 116, 115, 32, 99, 111, 114, 101, 44, 32, 99, 114, 121, 112, 116,
         111, 103, 114, 97, 112, 104, 121, 32, 105, 115, 32, 97, 98, 111, 117, 116, 32,
         109, 97, 107, 105, 110, 103, 32, 101, 110, 99, 114, 121, 112, 116, 105, 111,
         110, 32, 115, 121, 115, 116, 101, 109, 115, 32, 116, 104, 97, 116, 32, 99, 97,
         110, 32, 107, 101, 101, 112, 32, 115, 101, 99, 114, 101, 116, 115, 32, 97, 119,
         97, 121, 32, 102, 114, 111, 109, 32, 115, 109, 97, 114, 116, 32, 97, 110, 100,
         32, 119, 105, 115, 101, 32, 97, 116, 116, 97, 99, 107, 101, 114, 115, 32, 119,
         104, 111, 32, 116, 114, 121, 32, 116, 111, 32, 99, 114, 97, 99, 107, 32, 116,
         104, 101, 109, 46, 32, 32, 97, 32, 102, 97, 109, 111, 117, 115, 32, 110, 115,
         97, 32, 109, 97, 120, 105, 109, 32, 115, 116, 97, 116, 101, 115, 32, 116, 104,
         97, 116, 32, 97, 116, 116, 97, 99, 107, 115, 32, 110, 101, 118, 101, 114, 32,
         103, 101, 116, 32, 119, 111, 114, 115, 101, 44, 32, 116, 104, 101, 121, 32,
         111, 110, 108, 121, 32, 103, 101, 116, 32, 98, 101, 116, 116, 101, 114, 46, 10}
```

This now lists each characters as a number. The alphabet a-z itself is represented by the letters 97-122. The following command is Mathematica code that randomizes the letters, as a function “substcipherfunction”. “rp” is a random permutation that gets used. Please note that if you try this same command twice, you will almost certainly not get the same results each time. There are 26 factorial

```
In[11]:= 26!
Out[11]= 403 291 461 126 605 635 584 000 000
```

possibile permutations!

```
In[29]:= rp = PermutationList[RandomPermutation[26]] + 96;
         substcipherfunction[n_] := If[n ≥ 97 && n ≤ 122, rp[[n - 96]] - 32, n]
```

This next command then creates the cipher (as in class, I use lowercase for the plaintext, and UPPERCASE for the encrypted text):

```
In[35]:= cipherednumbers = Table[substcipherfunction[numbersofplaintext[[k]]],
           {k, Length[numbersofplaintext]}]; FromCharacterCode[cipherednumbers]
Out[35]= FS NSM YGZI, YZQXSGRZFXPQ NM FWGJS CFBNLR ILYZQXSNGL MQMSICM SPFS YFL BIIX
         MIYZISM FAFQ EZGC MCFZS FLD ANMI FSSFYBIZM APG SZQ SG YZFYB SPIC.  F
         EFCGJM LMF CFONC MSFSIM SPFS FSSFYBM LIKIZ RIS AGZMI, SPIQ GLTQ RIS WISSIZ.
```

Now, let’s try to decode this logically, without referring to the example that it was created from above. Here is a list of frequency of the letters from before, with 97=A and 122=Z:

```
In[32]:= Table[{ToUpperCase[FromCharacterCode[k1]],
           Count[Table[substcipherfunction[numbersofplaintext[[k]]],
             {k, Length[numbersofplaintext]}], k1]}, {k1, 65, 90}] // Transpose // TableForm
```

Out[32]//TableForm=

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
4	5	8	1	2	21	10	0	19	2	1	8	16	6	1	6	8	4

At this point I cannot say anything further for the general example, because it depends on the random permutation used to create the key. Thus if you run your own example (which I encourage you to do, to make sure you understand the material), what follows cannot help you.

However, for those looking at the webpage, I’ll continue with this example. We see that S and F are the most common letters, with I, M, and Z close by. In fact, the first world “FS” is made of the most common characters. The second sentence begins with “F”, so “F” is probably “a”, and then “S” is probably “t”.

```
In[40]:= replace[let_, with_] :=
          Do[If[cipherednumbers[[k]] == ToCharacterCode[let][[1]], cipherednumbers[[k]] =
             ToCharacterCode[with][[1]]], {k, Length[cipherednumbers]}]
```

```
In[42]:=
replace["F", "a"]; replace["S", "t"]; FromCharacterCode[cipherednumbers]
Out[42]=
"at NtM YGZI, YZQXtGRZaXPQ NM aWGJt CaBNLR ILYZQXtNGL MQMtICM tPat YaL BIIX MIYZItM aAaQ EZGC
MCaZt aLD ANMI attaYBIZM APG tZQ tG YZaYB tPIC.  a EaCGJM LMa CaONC MtatIM tPat attaYBM LIKIZ RIt AGZMI,
tPIQ GLTQ RIt WIttIZ.
"
```

Now the "P" in "tPat" screams out that it should be "h", so we make that change, too :

In[43]:=
```
replace["P", "h"]; FromCharacterCode[cipherednumbers]
```

Out[43]=
"at NtM YGZI, YZQXtGRZaXhQ NM aWGJt CaBNLR ILYZQXtNGL MQMtICM that YaL BIIX MIYZItM aAaQ EZGC MCaZt aLD ANMI attaYBIZM AhG tZQ tG YZaYB thIC. a EaCGJM LMa CaONC MtatIM that attaYBM LIKIZ RIt AGZMI, thIQ GLTQ RIt WIttIZ.
"

The "G" in "tG" is most probably an "o", so we ' ll try that :

In[44]:=
```
replace["G", "o"]; FromCharacterCode[cipherednumbers]
```

Out[44]=
```
at NtM YoZI, YZQXtoRZaXhQ NM aWoJt CaBNLR ILYZQXtNoL MQMtICM that YaL BIIX
  MIYZItM aAaQ EZoC MCaZt aLD ANMI attaYBIZM Aho tZQ to YZaYB thIC.  a
  EaCoJM LMa CaONC MtatIM that attaYBM LIKIZ RIt AoZMI, thIQ oLTQ RIt WIttIZ.
```

Consider the words "attaYBIZM" and "attaYBM". It seems likely to be attackers and attacks. In fact, I is very common in this text, and e is the most common letter in English. We' ll take this leap of faith (note that if the attacker has some idea of what the message is, they can be more confident in making choices like this) :

In[45]:=
```
replace["Y", "c"]; replace["B", "k"]; replace["I", "e"];
replace["Z", "r"]; replace["M", "s"]; FromCharacterCode[cipherednumbers]
```

Out[45]=
```
at Nts core, crQXtoRraXhQ Ns aWoJt CakNLR eLcrQXtNoL sQsteCs that caL keeX
  secrets aAaQ EroC sCart aLD ANse attackers Aho trQ to crack theC.  a
  EaCoJs Lsa CaONC states that attacks LeKer Ret Aorse, theQ oLTQ Ret Wetter.
```

At this point, the message is almost readible!

The lesson we have learned is that even though the key size here is enormously large, the system failed to mask important, nonrandom features of the English language: letter frequency and word structure. Had we supressed the spaces and punctuation, letter frequency would have still helped us crack this code, though it would have taken a litle more effort.

Thus a good cryptosystem really has to encrypt the same letter differently, or else it will be subject to the frequency attacks.